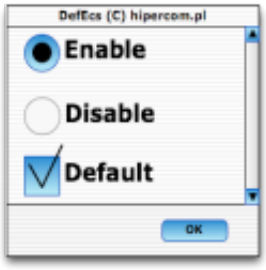


DefEcs v. 1.0 User Manual

Table of contents:

---

1. Introduction

DefEcs is a piece of software intended to be a helpful tool when browsing or modifying the Mac OS X defaults database. Why should anyone need to look into the system defaults? Well, for several reasons:

- curiosity (1) - it is always good to get familiar with your operating system,
- curiosity (2) - some people might like to know what information an application stores in the defaults database,
- cleanup - some applications leave their entries in the defaults system after uninstall (especially when uninstalling means just dragging the application bundle to the trash can); DefEcs makes searching and removing them easy,
- OS hacking - occasionally one may need to adjust some system defaults by hand (e.g. to turn a feature on or off), especially those which are not easily accessible from the system preferences panel,
- reviving an application - in some cases setting a certain combination of application preferences makes it impossible to run the application (e.g. to change the settings back to sensible values); it would be nice to have another way to change or reset the preferences in such cases,
- backups - DefEcs gives you an opportunity to export the current state of system defaults to an external file and, later, to load them back,
- development - it may prove helpful to have a tool to check and manipulate the preferences of an

application under development or testing (in fact that was the initial reason DefEcs was developed at all) - with DefEcs it is easy to check if the application under development stores its preferences in a proper form and place and if it reacts to changed preferences in the intended way,
- any other reason you may imagine when looking into system and application defaults may be helpful or educative.

Apple provides a command-line utility called **`defaults`**, which can be used to view and manipulate the system defaults as well. To get more information about this tool, open your terminal window and type **`man defaults`**.

The defaults database is organized as a set of *persistent domains* (or, shortly, *domains*). A domain is a *dictionary* (see also Arrays and dictionaries chapter of this manual) having a unique name. The name usually corresponds to the identifier of the application (for example **`pl.hipercom.DefEcs`**), which uses a particular domain for its purposes (in most cases the only purpose of an application domain is to save user preferences, window positions, recently visited directories and files etc.). A domain consists of *key-value* pairs. The *keys* or *key names* are just pieces of text indicating the meaning of the *values* they contain. Each value has its type (throughout the manual we will shortly say that a key has a type). Currently supported types are strings, numbers (boolean, integer and real), date/time values, binary data (a string of bytes without particular interpretation), arrays and dictionaries. Thus, a domain (which is a dictionary itself) may contain other dictionaries (or arrays), which in turn may contain more dictionaries (or arrays) and so on. The real (useful) data, however, is stored in the scalar values (strings, numbers, dates and binary data). Arrays and dictionaries serve only as containers to organize the data.

There are no restrictions on domain names except that they have to be unique (i.e. no two domains may have the same name). This rule applies to all dictionary keys - each key in a dictionary must be unique (must have a unique name).

There may be some domains not related to any application in particular. An example of such domain is **`NSGlobalDomain`** - a set of system-wide preferences, including your language, configuration of peripherial devices etc.

In this manual a *key* will denote a complete *key-value* pair, whereas by the *key name* we will mean the *key* part of the *key-value* pair. Thus, a sentence "domain `x.y.z` contains the key named `abc`, which is a dictionary" means, that in the domain `x.y.z` there exists a key-value pair, of which the key part is a name `abc` and the value part is a dictionary (possibly containing more keys, that is key-value pairs).

Back to the table of contents...

---

2. Warning

You should be extremely careful when manipulating the system defaults. It is strongly recommended to use DefEcs as a viewer only. Setting system defaults at random, without a thorough knowledge of what you're changing and why, may render your system and/or some applications unstable.

What the above means is that DefEcs may damage your system if used improperly. Like most tools, it may be helpful or destructive, depending on the way you use it.

Before editing any domains it is advised to export the current state of the defaults database to an external file (see Exporting and importing domains), so that you would have a stable version to restore the database should anything go wrong.

Please keep in mind, that although DefEcs comes as a universal binary for both Intel and PowerPC based Macs, it has been tested ***only on PowerPC architecture with Mac OS X 10.4*** (Tiger). According to the best knowledge of the author, it should work fine on the Intel platform with Mac OS X 10.4.x or later (as it is linked only against universal frameworks), but it is highly probable it will not work at all on Mac OS X versions prior to 10.4.

Back to the table of contents...

---

3. License agreement

By downloading and installing DefEcs you agreed to comply with the terms of the following license agreement.

NO SUPPORT

hipercom.pl does not provide any direct support for its free software. Any issues related to the free software manufactured by hipercom.pl should be resolved by consulting the User Manual, included in the software package, and visiting the product's web page.

Back to the table of contents...

---

4. The main window

The main window of the application consists of two horizontally adjacent panels. The left panel is the *domain list view*. It displays all persistent domains currently defined in the system defaults on the machine DefEcs is running on. The contents of the domain selected in the domain list view are displayed in the right-hand panel - the *domain contents view*. When you select another domain (either by clicking at its name with a mouse pointer or by selecting it in the table in any other way, e.g. using cursor keys), its contents are automatically displayed in the domain contents view, replacing the previously displayed domain.

The domain list view consists of three columns. The column marked as ***** (asterisk) indicates unsaved changes in a domain. If there is a little cross (**x**) by the name of a domain, that domain has been changed by the user and the new contents have not been saved to the system defaults database. The column named **Domain name** shows the names of all persistent domains existing in the system database when DefEcs last loaded it (either at the start of the application or when you requested reloading of domains). The third column, labeled **keys**, contains the number of top-level keys in a particular domain. By top-level keys we mean keys enclosed by the domain itself - the domain may contain much more keys if some of the top-level keys are containers (arrays or dictionaries). Above the domain list there is a search field. By typing a text in this field you make the domain list find and select a domain, whose name contains the entered string.

The right-hand panel of the main window displays the contents of the selected domain. This panel is described in more detail in the section The domain contents.

You can modify the sizes of the columns by dragging the border between column headers left or right. You can also change the order, in which the columns are displayed, by dragging the header of a column and placing it in-between other columns. The size of the domain list view and domain contents view can be changed by dragging the splitter knob (a little ball in the vertical divisor between the views) left or right. Of course if you resize the whole window, the views will resize themselves accordingly.

Back to the table of contents...

---

5. The domain contents

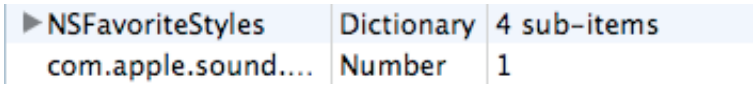
A *domain* consists of a set of *keys*. By a key we mean a value (or another set of values) indicated by either a name (if the value is an item of a dictionary) or an index (if the value is an item of an array). The domain is a dictionary, therefore each top-level key has to have a unique name within the domain. Any non-empty string can be a key name. Each key (i.e. a name-value or index-value pair) has a *type*.

The type defines what kind of data can be stored in the key. The supported data types are:

- numbers - boolean, integer and real values,
- dates - a date and/or time values,
- strings - some text (in general unicode characters),
- binary data - a chain of bytes without particular interpretation (usually data values contain encoded objects),
- arrays - containers able to store a set of keys, each of which has a unique index (ordering number),
- dictionaries - containers able to store a set of keys, each of which has a unique name (string).

A type once assigned to a key cannot be changed. To change the type of a key, you have to delete the key and re-create it with a different type.

The domain contents view represents a tree structure of the domain. Each container (an array or a dictionary) can be *expanded* to view its contents. If some of the keys in a container are containers themselves, they can be expanded as well, and so on. To expand a container, double-click its name or single-click a little triangle next to the name of the key.



The key **NSFavoriteStyles** is collapsed.

The same key in the expanded state.

In the example above, the key **NSFavoriteStyles** is a dictionary. It contains 4 keys named **Italic**, **Outlined**, **Shadowed** and **Bold**, which are dictionaries themselves, and as such can be further expanded. The key **com.apple.sound.beep.feedback** (the name truncated to **com.apple.sound...** in the example above), on the other hand, is a number (not a container) and cannot be expanded.

The domain contents view has three columns. The first column displays the key name or index, the second column shows the type of the key, and the third one shows either the value of the key, or the number of sub-keys if the given key is a container. In the example above, the key **NSFavoriteStyles** has 4 sub-keys, each of which is a dictionary containing a single sub-key.

Back to the table of contents...

---

6. Arrays and dictionaries

To understand the behaviour and some limitations of DefEcs it is important to learn the properties of an array versus a dictionary. Both objects serve as containers for other objects, i.e. they don't contain any valuable data themselves (only other keys, which may hold values or be another containers), and the main difference between them is the way the inner objects are identified.

Each object contained in an array has a unique index - a number indicating the position of the object in the array. The indices must be both unique and continuous, starting from 0 and going through 1, 2, 3 etc. In DefEcs, when you add a new key to an array, it is always put at the end of the array (and unfortunately in the current version there is no way to reorder the array items other than removing the

keys and add them again in a different order).

In a dictionary, on the other hand, each value is identified by a name rather than an index. A name is just a piece of text. However, names in a dictionary must be unique (that is, no two objects can have the same name).

The items of an array are always displayed in the order based on their indices. In case of dictionaries, the order of items is irrelevant. Currently DefEcs displays them in the order their names are returned by the enclosing dictionary. Ordering of items based on the preferences of the user is a feature planned in the next version of DefEcs.

Back to the table of contents...

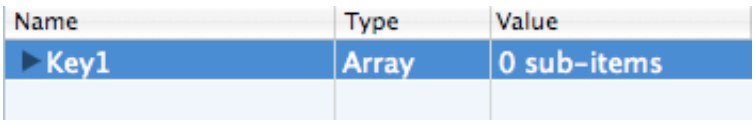
---

7. Adding a new key

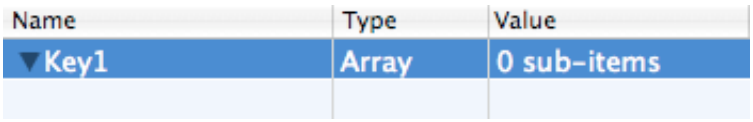
To add a new key to the currently selected container select **New key** from the **Key** menu or press **Command-N**. The question is: which container will be the new key added to? The rules below apply:

- if a domain is empty, the new key will be a top-level key in the domain,
- if the currently selected item in the domain contents view is a sub-item of a container and that item is not a container itself, the new key will be added to that container,
- if the currently selected item is a container and it is expanded, the new key will be added to that container,
- if the currently selected item is a container and it is collapsed, the new key will be added to the container enclosing the selected item (if the selected item is a top-level container, the new key will be added to the domain itself).

Suppose you have a domain with a single key in it, being an empty array. If you add a new key in the collapsed state (as on the picture below)

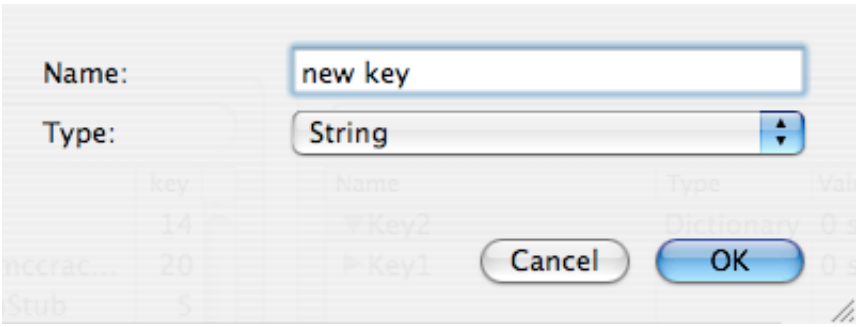


the new key will be added as a top-level key in the domain. If, on the other hand, you first expand the array **Key1** (as on the picture below - look at the direction of the triangle by the name of the key: it points downwards now)



the new key will be added as the first sub-item (i.e. the item number 0) of the array **Key1**. The same rule applies to all levels of containers.

After selecting the **New key** action you will be asked for the name of the key (if adding a key to a dictionary) and its type.



If you are adding the key to an array, the **Name** field will be inactive and will be displaying the index of the new item. When the key type is **Array** or **Dictionary**, the key is immediately added after clicking the **OK** button. If you selected any other type, after clicking OK you will be presented with another sheet allowing you to set the value of the new key (see Modifying the value of a key). If, at any moment, you press the **Cancel** button, the new key will be discarded and no changes to the domain will be made.
If adding a key to a dictionary, it is not allowed to leave the key name empty. In that case the key will not be added and you will be presented with an error message. Similarly, if you enter a name already existing in the dictionary the key is being added to, an error message will be displayed and the operation will be canceled.
After adding a key, the domain is considered to be changed, thus a cross will appear next to the domain name in the domain contents view. The new key is not stored in the system database until you save changes made to the domain (see also Saving changes), unless you selected **Save all changes immediately** in the DefEcs preferences (see also The user preferences).

Adding a new key can be undone (see Undoing operations).

Back to the table of contents...

---

8. Deleting a key

To delete a key first select it in the domain contents view. You can delete a single item or a whole container. However, it is not possible to delete all top-level keys from a domain in a single step. You have to point and delete each key separately. In case of all other (not the top-level) containers, selecting a container and deleting it removes the container with all its contents.
To delete a key, select **Delete key** from the **Key** menu or press **Command-D**. After confirming the deletion, the key will be removed and the domain contents view updated accordingly.
The key is not physically removed from the system database until you save changes made to the domain (see also Saving changes), unless you selected **Save all changes immediately** in the DefEcs preferences (see also The user preferences).

Deleting a key can be undone (see Undoing operations).

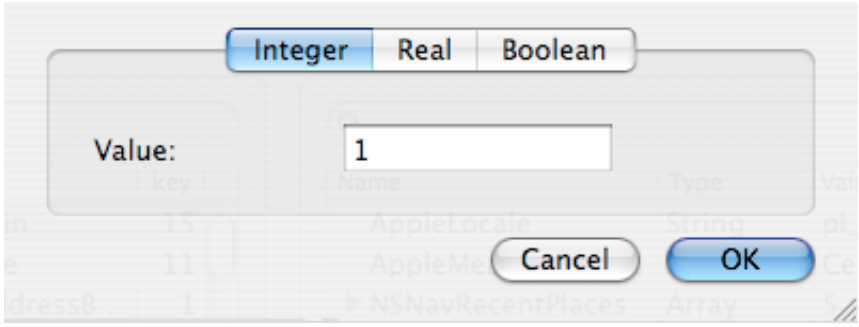
Back to the table of contents...

---

9. Modifying the value of a key

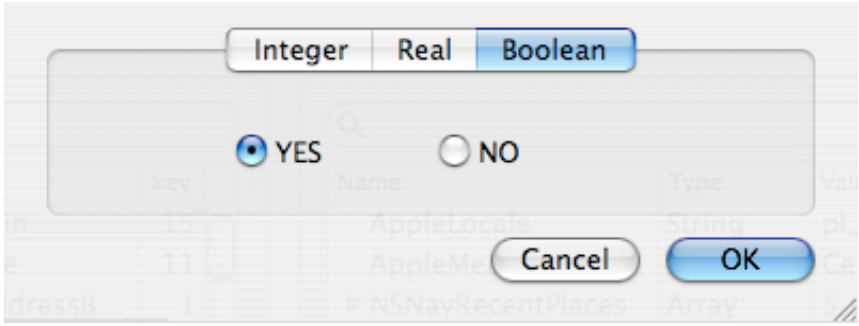
To modify the value of a key you have to double-click the key with your mouse in the domain contents view. Note, that you can not edit containers. Double-clicking on a container makes it expand or collapse. Each key type has an associated editing sheet. Strings, numbers, dates, binary data - they are all edited in a different way. When you edit a string value, the sheet allows you to enter a line of text. The current key

value is always displayed in the edit field, ready to be changed or replaced. As in most applications, copy-pasting text from other windows works with the string edit sheet.
When you choose to edit a number value, you are presented with a sheet shown below.



The number edit sheet.

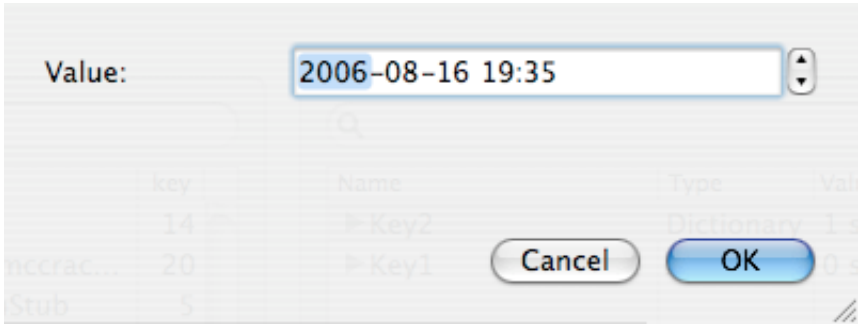
The number value can be either integer, real or boolean (logical). They are all numbers, although boolean values are labeled **Boolean** in the domain contents view, whereas integer and real values are labeled **Number** (technically, they are all objects of class `NSNumber` or `NSCFNumber` or one of its descendants, like `NSCFBoolean`). When you double-click a number value, the editing sheet will set the number type according to the value currently stored in the number (i.e. when it is a boolean value, the **Boolean** tab will be active, when it is not a boolean value and has no fractional part the **Integer** tab will be active and the **Real** tab will be active when the number has a non-zero fractional part).
The new type of a number is determined depending on which tab is active in the moment you click the **OK** button. Thus, the numeric value can be saved as a boolean value if you leave the **Boolean** tab active when you click **OK**.



The boolean value edit sheet.

All non-zero numerical values mean *YES* in the boolean context, while zero means *NO*. On the other hand, a *YES* boolean value has the numerical value of 1. However, when displaying a boolean value in the domain contents view, DefEcs translates numerical values into YES or NO boolean values. Nevertheless, keep in mind that each YES in your domain is in fact a 1, and each NO is a 0.
Moreover, if you save a real value with no fractional part, the next time you edit it the **Integer** tab will be active (DefEcs distinguishes between integer and real numbers by checking their fractional parts).

When you edit a date value, you will be presented with a date/time picker.

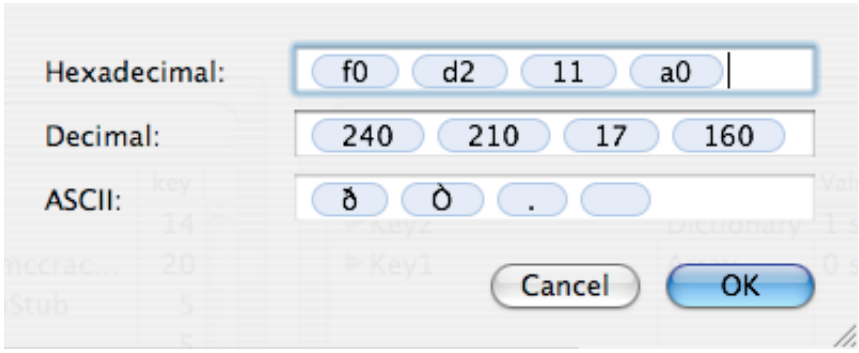


The date/time edit sheet.

Technicaly, the date/time values are stored in the system defaults database as objects of type

`NSCFCalendarDate`, capable of storing both date and time. If the time is meaningless in the particular context, you can safely leave the time part of the value set to 00:00. Instead of typing the date in, you can highlight the part of the date and/or time and use the up/down buttons (next to the edit field) to set the desired value.

The last editable key type is the binary data. It is just a chain of bytes (a *byte* is a numerical value from 0 to 255 decimal or from 00 to ff hexadecimal).



The binary data edit sheet.

Each byte is represented by a small blue oval with the numerical value inside it. You can edit the values only in the **Hexadecimal** and **Decimal** fields. The **ASCII** field shows the interpretation of bytes as characters (although considering that binary data usually contains encoded objects or values of custom types, the ASCII representation is usually meaningless). After typing in a value in one of the editable fields you have to ***press the space bar*** to make the value appear as another byte (blue oval) and to update other fields (that is, entering a hexadecimal number and pressing space causes that number to be interpreted both as decimal and ASCII and updates the other edit fields).
To remove a byte from the chain, just highlight it and press **Delete** on your keyboard.

The new value of a key is not stored in the system database until you save changes made to the domain (see also Saving changes), , unless you selected **Save all changes immediately** in the DefEcs preferences (see also The user preferences).

Changing the value of a key can be undone (see Undoing operations).

Back to the table of contents...

---

10. Saving changes

As was mentioned earlier, the changes made to a domain are not physically stored in the system database until you save changes made to a domain. You can save a single domain (by choosing **Save domain** from the **Domain** menu or pressing **Command-S**) or save all domains in a single step (by choosing **Save all domains** from the **Domain** menu or pressing **Command-Shift-S**).
It is not allowed to *save an empty domain*, i.e. a domain with no keys at all (that rule does not apply to other empty containers inside a domain). If you try to do this, you will be shown an error message

Please understand that this restriction is introduced by the system defaults, not by DefEcs. Physically, saving an empty domain is equivalent to removing that domain from the system database. Therefore, if you try to save an empty domain, no action is performed and if the domain was marked as modified (with a cross next to its name in the domain list view), it remains marked (as no changes were actually stored in the system database).
Similarly, if you choose to save all domains in a single step and at least one domain is empty, you will be presented with a warning similar to the one shown above and only non-empty domains will be saved to the system defaults database. Any changed empty domains will remain marked as changed.
Note, that saving changes to the system defaults cannot be undone. However, even after saving a domain you can undo changes made to that domain and save it again in its previous state (see Undoing operations).
If you turn on the **Save all changes immediately** feature in the preferences panel (see The user preferences), saving domains by hand is not necessary. After each operation the modified domain will be saved automatically. The only exception is an empty domain (see above). If you add a new (empty) domain, it will be auto-saved after you add the first key to it.

Back to the table of contents...

---

11. Reloading domains

If you have made changes to a domain and you wish to restore its contents from the system database, you can *reload a domain*. Reloading discards all changes made to a domain and loads its contents from the on-disk database. If you attempt to reload a changed domain (marked with a cross), you will be warned that all changes made to this domain will be lost. A good reason to reload a domain (or all domains) is to keep the contents of displayed domains up to date with the system database. As you're looking at and editing values in a snapshot of the database, some other applications may change the system defaults as well. Thus, you may wish to reload domains periodically to make sure you edit the most recent version.
To reload a single domain, select it in the domain list view and choose **Reload domain** from the **Domain** menu or press **Command-R**. To reload all domains either choose **Reload all domains** from the **Domain** menu, or press **Command-Shift-R**.
Note, that by reloading all domains you also reload the list of available domains. Therefore, if you had added a new domain and had not saved it before you decided to reload all domains, the new domain is discarded (as it is not yet present in the defaults database).
All reloaded domains are marked as non-changed. Reloading a domain or domains *clears the undo stack* (i.e. no **Undo** operation is possible after reloading domains), so please take care when using this feature.

Back to the table of contents...

---

12. Adding a new domain

To add a new domain to the system database, select the **New domain** item from the **Domain** menu or press **Command-Shift-N**. You have to enter the name of the new domain. Please remember, that the name must be unique and cannot be empty. In such cases an error message will be displayed and the domain will not be added.
The new domain is not stored in the system database until you save changes made to the domain (see also Saving changes). Please note, that an empty domain cannot be saved, so after adding a new domain it would be wise to add at least one key to the domain before saving it.
Adding a new domain can be undone (see Undoing operations).

Back to the table of contents...

---

13. Deleting an existing domain

To remove a domain with all its contents from the defaults database, either select **Delete domain** from the **Domain** menu, or press **Command-Shift-D**. After confirming your choice the domain will be removed. This operation is immediately stored in the defaults database (there is no need to save the changes, especially that there is no domain to save as you have just deleted it).
Although when you remove a domain the changes are immediately saved to the defaults database, this operation can still be undone (see Undoing operations) and the removed domain will be added back. However, after adding the domain back via the **Undo** operation, you have to save changes to the restored domain (see also Saving changes).

Back to the table of contents...

---

14. Undoing operations

Most operations available in DefEcs can be undone and then redone again, but not all of them. To undo the last operation, choose **Undo [operation name]** from the **Edit** menu or press **Command-Z**. An undone operation can be redone via the **Redo [operation name]** item from the **Edit** menu or by pressing **Command-Shift-Z**.
The operations that can be undone are:

- adding a key,
- deleting a key,
- changing the value of a key,
- adding a domain,
- deleting a domain.

The operations that cannot be undone are:

- reloading a domain,
- reloading all domains,
- importing the contents of the database from an external file.

Please note, that no operation can be undone (even one of those mentioned above as possible to undo) after you reload a domain or all domains from the defaults database. The reason is that if you reload a domain, in which you changed a key, and there is no such key in the reloaded version, changing the key back to its previous value makes no sense. To avoid such situations, reloading clears the undo stack.
After you undo an operation, the previous one is available for undoing and so on, until you reach the bottom of the undo stack, i.e. until you undo the first operation made after the undo stack had been cleared.

Back to the table of contents...

---

15. Exporting and importing domains

DefEcs allows you to export the current state of the defaults database to an external file. It seems a good idea to export the database before making any changes to it, so that you can restore the state of the database if anything goes wrong. To save the defaults database to a file select **Export all to file** from the **File** menu or press **Shift-Command-E**. You will be asked to choose the folder and enter the file name. The default file extension, appended to the name if you don't provide your own extension, is `.defdoms`. The format of the file is one created by the so called "keyed archiver" from the Cocoa framework (a hack: if you change the extension to `.plist`, you'll be able to open it with the system property editor, although it won't look exactly the same as the defaults database).

To import the database from the previously created file, select **Import all from file** from the **File** menu or press **Shift-Command-I**. After choosing the file, the domains stored in it will replace the domains being currently displayed and edited.

Please note, that importing from a file discards all unsaved changes.

After you import the contents of domains from an external file, all loaded domains will be marked as changed and the loaded contents will not be stored in the system database until you save the changes (see also Saving changes), even if you selected auto-saving in the preferences.

It is also possible to export and import a single domain. This feature is useful if you plan to manipulate keys in a certain domain and want to make a back up copy of it (in such cases backing up the whole database is not the best solution, because when you load it back, you will loose also the changes made to all other domains, and such changes may be made in the meantime by other applications).
To export a single domain, select it in the domain list view and choose **Export to file** from the **File** menu (or press **Command-E**). The default extension of the file containing a single domain is `.defdom`.
To import a domain, choose **Import from file** from the **File** menu or press **Command-I**. The imported domain will be marked as changed and the loaded contents will be put into the defaults database after you save the domain (see also Saving changes).

Back to the table of contents...

---

16. The user preferences

To display the DefEcs user preferences window, select **Preferences** from the **DefEcs** menu or press **Command-,** (comma). You can control the behaviour of DefEcs in the following aspects:

- **Save all changes immediately**
  If selected, all changes made to a domain will be saved to the system database automatically. Saving changes via **Command-S** is not necessary. However, if you add a new (empty) domain or remove the last key from a domain (thus making it empty), the changes will not be saved. Empty domains are not accepted by the defaults database (see also Saving changes).
- **Domain search**
  If **Substring in any place** is selected, DefEcs will select the first domain containing the search string as a part of its name. On the other hand, if you select **Substring at the beginning**, DefEcs will select a domain only if its name starts with the search string.
- **Key search**
  The options **Substring in any place** and **Substring at the beginning** have the same meaning as in Domain search. The last check box - **Search also in values** - lets you decide whether a key should be selected only if its name matches the search criteria (unchecked), or either the name or the value does (checked). If you select **Substring at the beginning** and **Search also in values** simultaneously, a key will be selected only when its name combined with the value (more

precisely, the value appended at the end of the name) starts with the search string.

The button **Factory defaults** restores the default settings.

All changes made in the preferences window are applied immediately - no saving is required. To close the preferences window, use the standard close button on the title bar.

Back to the table of contents...

---

17. Frequently asked questions

Q:
What the name DefEcs stands for?
A:
Def stands for Defaults and Ecs stands for X, as in OS X.

Q:
Do I really need this application?
A:
I don't know. Do you? A hint: if you've ever used the `defaults` command line utility, you will probably find DefEcs helpful.

Q:
Do you provide any support for DefEcs?
A:
According to the license agreement - no. However, you may send questions and suggestions to `defecs@hipercom.pl` and the most frequently asked questions will be answered in the FAQ on the web page (and some of them added to the FAQ in the help book of the next release).

Q:
I have a version of Mac OS X prior to 10.4. Can I use this software?
A:
Probably not. Due to the lack of time and resources, we have tested this software only on Mac OS X 10.4 (on a PowerPC G4). It is a properly compiled universal binary, so it should also work on an Intel based Macs with OS X 10.4 or later, but it is very unlikely it will work on OS X versions prior to 10.4 (maybe an updated 10.3.9...).

Q:
How do I uninstall DefEcs?
A:
First, use DefEcs to remove the domain **`pl.hipercom.DefEcs`** from your system defaults (if present). Then drag **DefEcs.app** to your trash can.

Q:
I'd like to add a value of the class MyCustomClass but I'm not able to. How come?
A:
The defaults database only supports the basic types listed in DefEcs' **New key** sheet. All other objects are first encoded to binary data type and then added as such (well, it's a bit more complicated but let's stay with the simplified version). Some other objects encode themselves to dictionaries. So, if you are a developer and wish to add an object of your class to the defaults, use the representation suitable for your `initWithCoder:` (or equivalent in other languages) class initializer.

Q:
Is there a new version of DefEcs planned to be released?
A:
Yes, visit the web page www.hipercom.pl to read about the planned features. Maybe the new version is ready even as you're reading it.

Q:
I have a backup copy but I need to restore only a single domain from it. Is it possible?
A:
Yes. First, turn off the **Save all changes immediately** feature in DefEcs preferences (if turned on) and import domains from the copy you have. All domains will be imported and marked as changed. Now, select and save the domain you wish to restore (do not save all domains, just the one you need to get restored). Quit DefEcs, selecting **Don't save** when it informs you about unsaved changes. That's it - only a single domain restored. Of course a better solution would be to export just a single domain before making any changes to it, but, according to the solution provided above, even if you export the whole database, you are still able to restore only selected domains from the backup copy.

Back to the table of contents...
